Paradigmas de la Programación – Segundo Parcial - Primera Fecha

14 de Junio de 2022

Apellido y Nombre: _____

[20 pt.] ¿Qué características de los lenguajes de scripting se pueden observar en el siguiente programa? Siempre que sea posible, cite los fragmentos específicos de código en los que se observan las propiedades que mencione.

```
tell application "Finder"
    set passAns to "app123"
    set userAns to "John"
    if the text returned of (display dialog "Username" default answer "") is userAns then
        display dialog "Correct" buttons {"Continue"} default button 1
        if the text returned of (display dialog "Username : John" & return & "Password" default answer ""
buttons {"Continue"} default button 1 with hidden answer) is passAns then
            display dialog "Access granted" buttons {"OK"} default button 1
        else
            display dialog "Incorrect password" buttons {"OK"} default button 1
        end if
    else
        display dialog "Incorrect username" buttons {"OK"} default button 1
    end if
end tell
```

[20 pt.] El siguiente código forma parte de un código de framework. ¿Qué tipo de elemento vamos a encontrar dentro de la etiqueta Router¿

```
import { BrowserRouter as Router } from 'react-router-dom'

const App = () {
  render() {
    return (
        <Router>
           // your routes here
        </Router>
    )
  }
}
```

[20 pt.] A partir de la siguiente base de conocimiento, ¿qué subobjetivos se tienen que satisfacer para probar que es cierto que `alivia(aspirina,gripe).`?

```
enfermo_de(manuel,gripe).
tiene_sintoma(alicia,cansancio).
sintoma_de(fiebre,gripe).
sintoma_de(tos,gripe).
sintoma_de(cansancio,anemia).
elimina(vitaminas,cansancio).
elimina(aspirinas,fiebre).
elimina(jarabe,tos).
recetar_a(X,Y):-enfermo_de(Y,A),alivia(X,A).
alivia(X,Y):-elimina(X,A),sintoma_de(A,Y).

enfermo_de(X,Y):-tiene_sintoma(X,Z),sintoma_de(Z,Y).
```

[10 pt.] Según el siguiente texto:

> *Hewitt argued against adding the requirement that messages must arrive in the order in which they are sent to the actor. If output message ordering is desired, then it can be modeled by a queue actor that provides this functionality. Such a queue actor would queue the messages that arrived so that they could be retrieved in FIFO order. So if an actor X sent a message M1 to an actor Y, and later X sent another message M2 to Y, there is no requirement that M1 arrives at Y before M2. In this respect the actor model mirrors packet switching systems which do not guarantee that packets must be received in the order sent. Not providing the order of delivery guarantee allows packet switching to buffer packets, use multiple paths to send packets, resend damaged packets, and to provide other optimizations.*

¿con qué tipo de mensajes se comunican los actores?

[30 pt.] Comente el siguiente texto, comparando las estrategias de manejo de excepciones y la filosofía "let it crash". En su justificación, incorpore los principios de diseño de resiliencia y elasticidad propios de la orientación a actores. Mencione cómo se relaciona la programación defensiva con una filosofía "let it crash".

Si lo desea, puede rendir este ejercicio mediante un comentario oral.

> *Erlang is designed with a mechanism that makes it easy for external processes to monitor for crashes (or hardware failures), rather than an in-process mechanism like exception handling used in many other programming languages. Crashes are reported like other messages, which is the only way processes can communicate with each other, and subprocesses can be spawned cheaply. The "let it crash"philosophy prefers that a process be completely restarted rather than trying to recover from a serious failure. Though it still requires handling of errors, this philosophy results in less code devoted to defensive programming where error-handling code is highly contextual and specific.*